

Estrutura de um Sistema Operacional

Introdução

- Sistema Operacional: conjunto de rotinas que oferecem serviços aos usuários, às suas aplicações e ao próprio SO.
 - Tais rotinas se encontram no Núcleo do Sistema (*Kernel*).
 - Esse *Kernel* deve ser bem protegido.
 - Proteção se dá por meio do conceito de chamada de sistema (System Call).
 - Usuário não deve ter acesso direto ao hardware.

Introdução

- Estrutura pode ser vista em camadas:
 - Aplicativos estão no mais alto nível.
 - Kernel está no mais baixo nível entre os programas.
 - Interage com o hardware

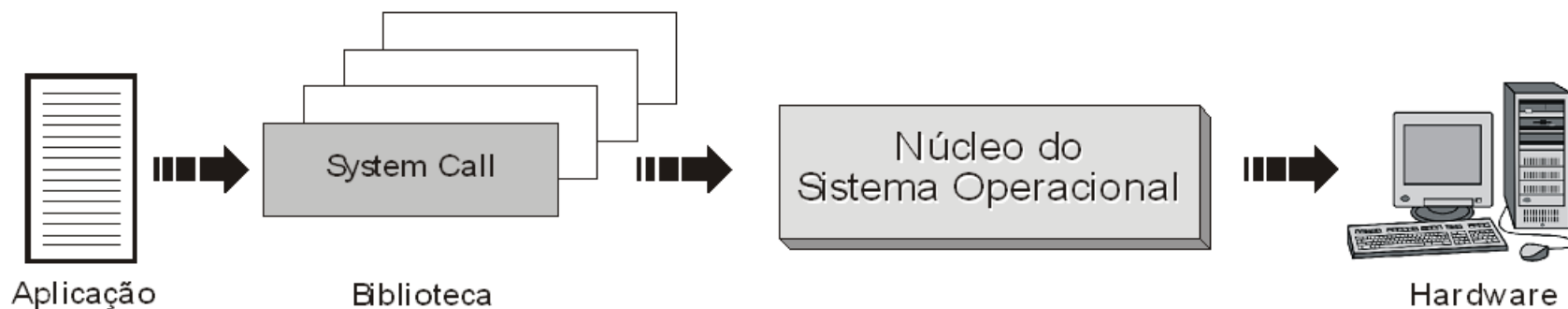


Modos de Acesso

- Modo Usuário (*User mode*):
 - Modo em que se encontram as aplicações do usuário.
 - Apenas instruções não privilegiadas podem ser executadas;
- Modo Supervisor (*Kernel mode*):
 - Modo em que se encontram as chamadas de sistema do Sistema Operacional.
 - Nesse modo podem ser executadas instruções privilegiadas e não privilegiadas.

System Call

- Porta de entrada para o núcleo do S.O.
 - Nomes: Call (Unix), System services (OpenVMS), Application Program Interface – API (Windows)
- Sempre que uma aplicação necessita interagir com o hardware ou executar uma instrução privilegiada, uma chamada de sistema é acionada.



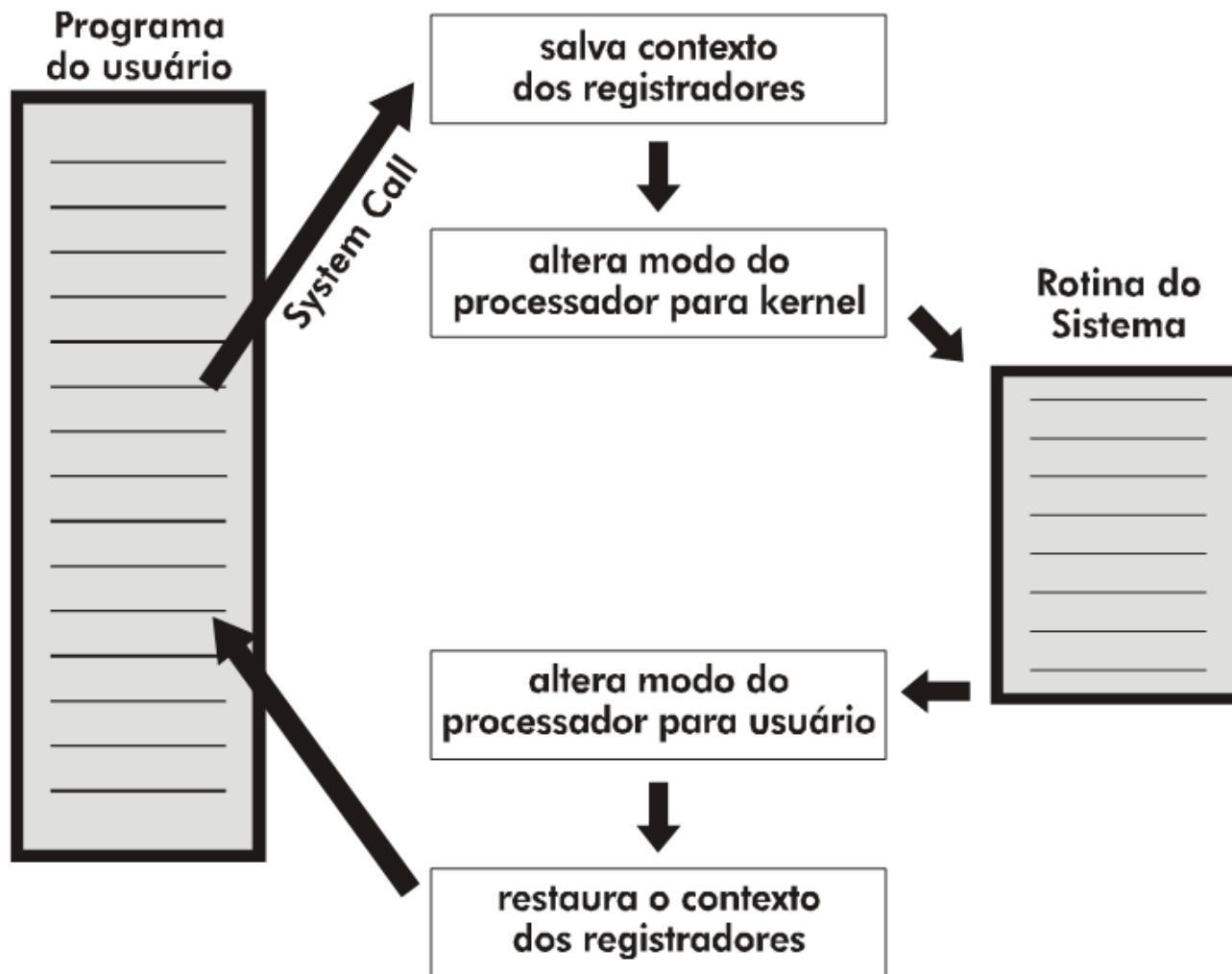
System Call

- Algumas das funções das chamadas de sistemas (system call):
 - Gerência de processos e threads
 - Gerência de memória
 - Gerência de dispositivos
 - Gerência de Sistemas de Arquivos

System Call

- Toda interação do programa do usuário com o hardware é monitorada:
 - Proteção por software: verifica se o usuário tem permissão para a execução de uma determinada *system call*.
 - Proteção por hardware: instruções privilegiadas só podem ser executadas pelo *kernel*.

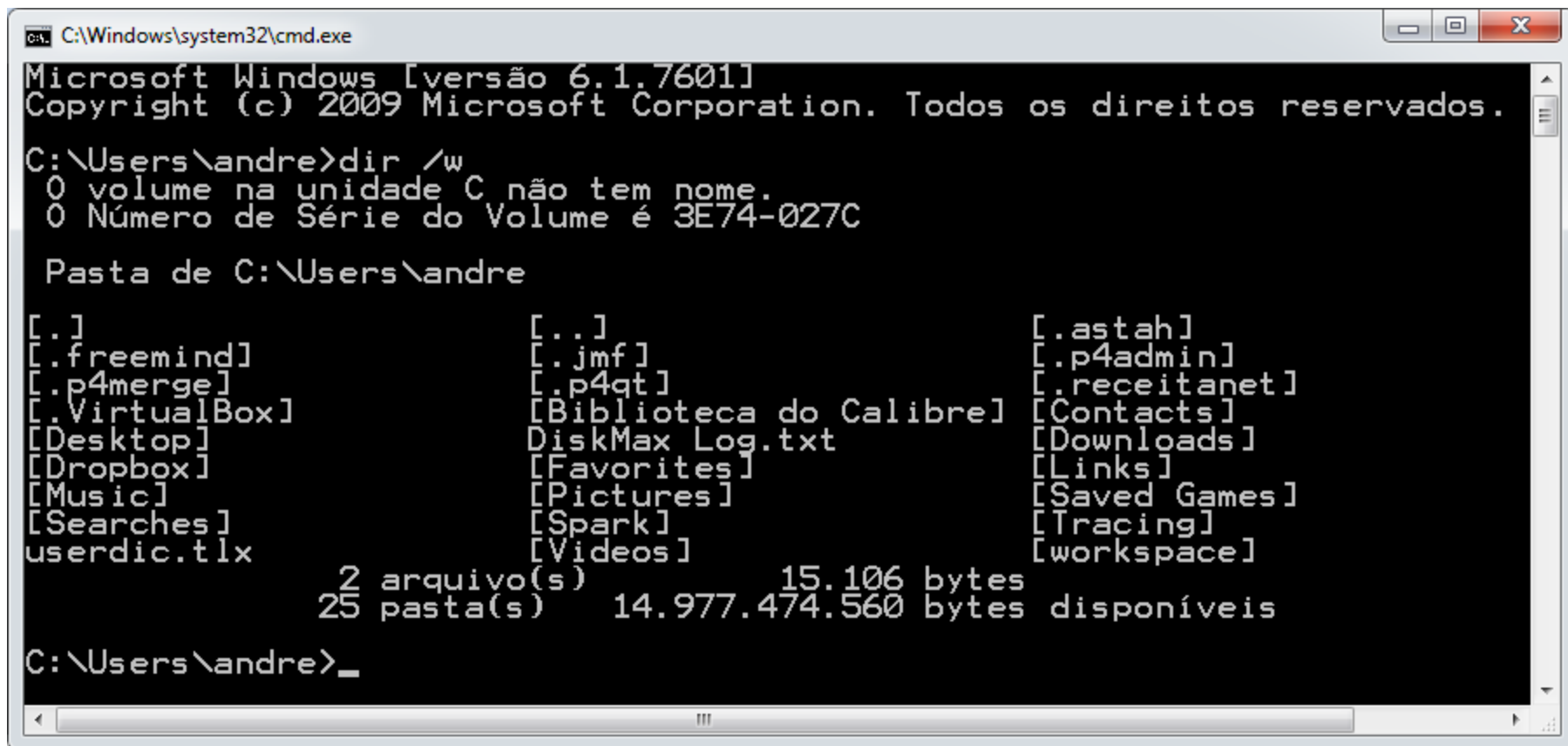
System Call



Interação com o Usuário

- Interação com o Sistema Operacional pode ocorrer por:
 - Interface Gráfica
 - Interface de linha de comando
 - Comandos são enviados para o Interpretador de Comandos
 - Interpretador de comandos executa as funcionalidades por meio do kernel
 - Resposta é apresentada ao usuário por meio de listagem de texto

Linguagem de Comandos



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versão 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Todos os direitos reservados.

C:\Users\andre>dir /w
 0 volume na unidade C não tem nome.
 0 Número de Série do Volume é 3E74-027C

Pasta de C:\Users\andre

[.]                [..]                [.astah]
[.freemind]        [..jmf]                [.p4admin]
[.p4merge]         [..p4qt]                [.receitanet]
[.VirtualBox]      [Biblioteca do Calibre] [Contacts]
[Desktop]          DiskMax Log.txt         [Downloads]
[Dropbox]          [Favorites]             [Links]
[Music]            [Pictures]              [Saved Games]
[Searches]         [Spark]                 [Tracing]
userdic.tlx        [Videos]                [workspace]

                2 arquivo(s)                15.106 bytes
                25 pasta(s)                14.977.474.560 bytes disponíveis

C:\Users\andre>
```

Ativação do Sistema

- Boot
 - Boot loader ativa o POST
 - Boot loader procura dispositivo com o Sistema Operacional
 - Sistema Operacional é carregado para a Memória Principal
 - Gerência do hardware passa para o Sistema Operacional

Arquiteturas do Núcleo

- Projeto de S.O. é complexo, requisitos podem ser conflitantes:
 - Confiabilidade
 - Portabilidade
 - Fácil Manutenção
 - Flexibilidade
 - Desempenho
- Escolha depende dos objetivos do sistema

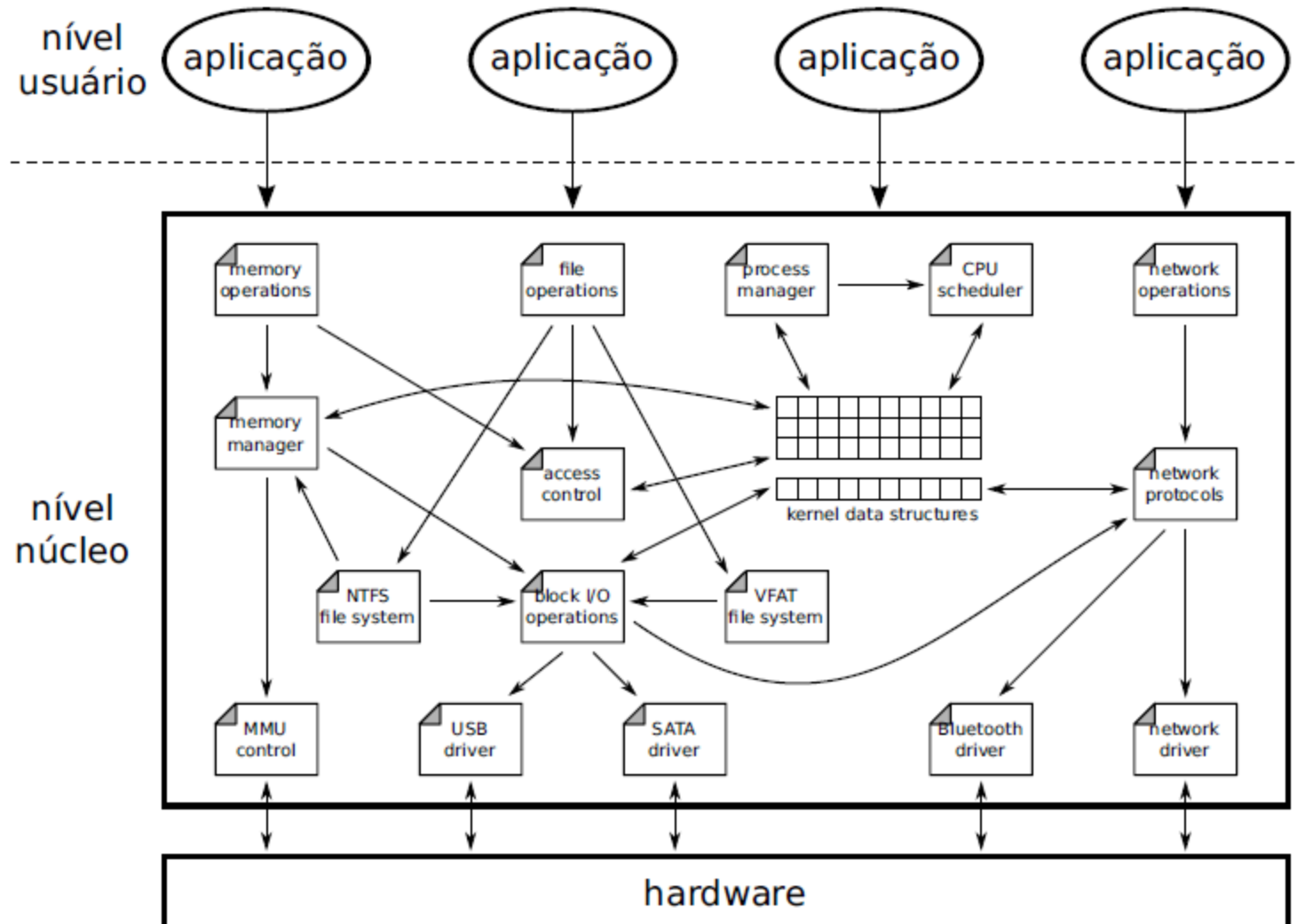
Arquiteturas do Núcleo

- Primeiros Sistemas Operacionais foram escritos em *Assembly*.
 - Milhões de linhas de código para dar manutenção.
- Sistemas atuais são escritos em linguagem Orientada a Objetos (C++).
- Em relação ao projeto do sistema, diversos tipos de arquitetura podem ser seguidos.

Arquitetura Monolítica

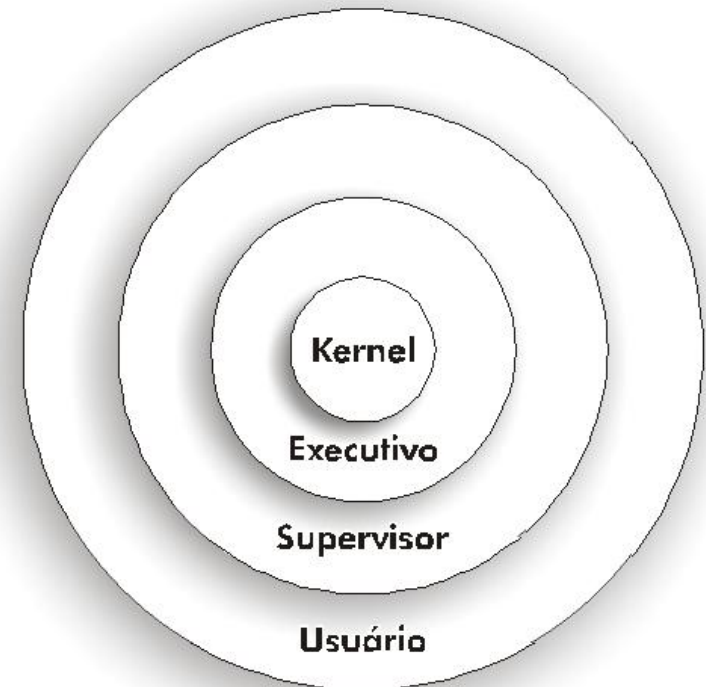
- Núcleo construído sem muita organização.
- Bloco de funcionalidades que se comunicam diretamente.
- Maior desempenho, mas pouca organização.
- Exemplo:
 - Sistemas embarcados.
 - Linux, primeiras versões.

Arquitetura Monolítica



Arquitetura em Camadas

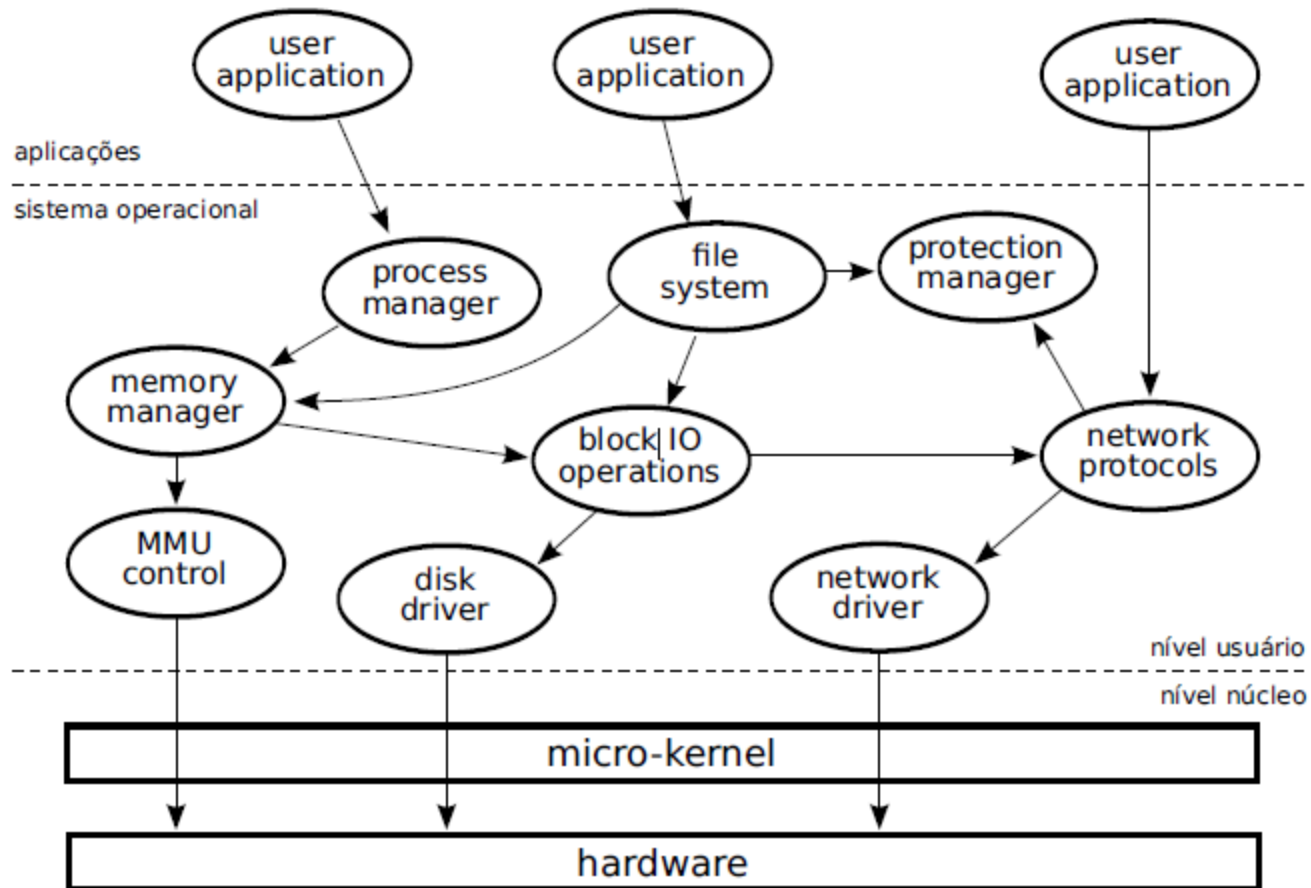
- Sistema é dividido em níveis sobrepostos.
- Funções do sistema operacional ficam isoladas.
- Melhor organização:
 - permite melhor depuração e manutenção



Arquitetura *Microkernel*

- Arquitetura cliente/servidor
- Utilização de troca de mensagens para comunicação entre processos
- Independência de funcionalidades
 - Erro em um servidor não para o sistema completamente
- Núcleo menor: mais fácil de manter e depurar
- Exige muitas trocas de modo de acesso
 - *Usuário* \leftrightarrow *Kernel*

Arquitetura Microkernel



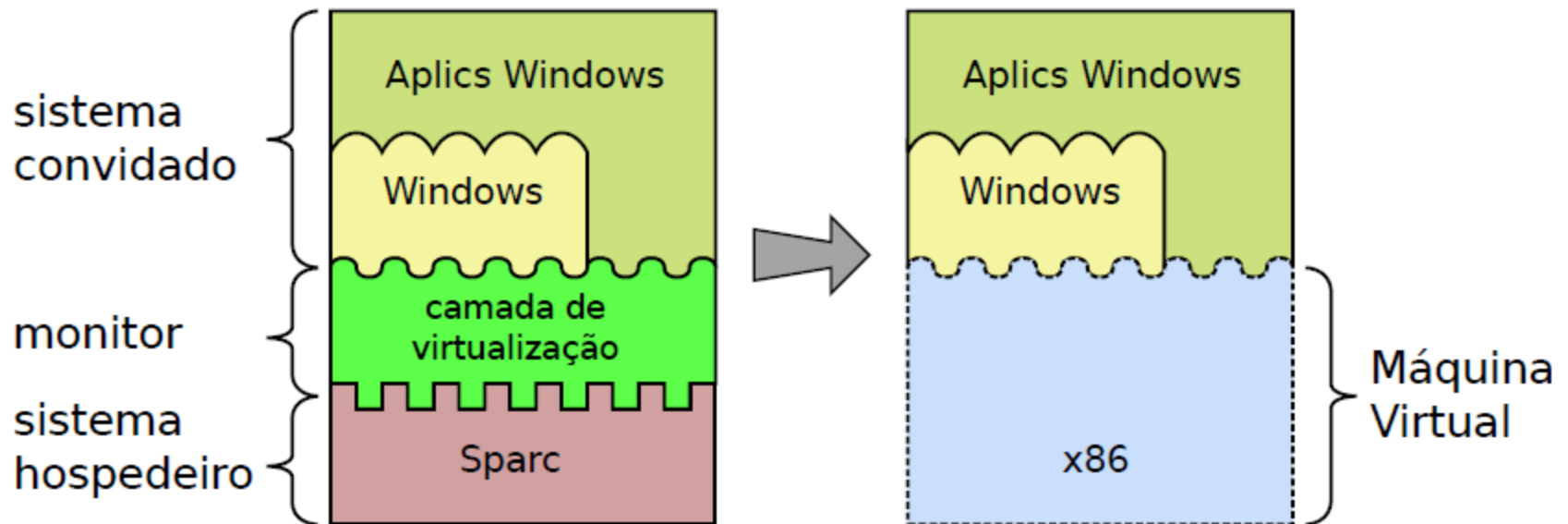
Virtualização - Motivação

- Arquiteturas atuais não são flexíveis
 - Não é possível criar novas instruções para (ou alterar) um processador.
 - Não é possível criar novas chamadas de sistemas.
 - Sistema Operacional é gerado especificamente para uma arquitetura.
 - Programas e bibliotecas gerados para uma arquitetura só funcionam na mesma.

Virtualização

- Permite a inclusão de uma camada de software entre o hardware e o Sistema Operacional
 - Permite então resolver os problemas de compatibilidade entre aplicações e arquiteturas.
 - Permite o acoplamento entre interfaces distintas.

Virtualização



Virtualização – Partes básicas

- Sistema Real ou Hospedeiro (*host system*):
 - Contém os recursos reais de hardware e software do sistema;
- Sistema Virtual ou Convidado (*guest system*):
 - Executa sobre o sistema virtualizado.
 - Às vezes, vários sistemas podem coexistir.
- Hipervisor ou Monitor de Virtualização (*VMM – Virtual Machine Monitor*):
 - Camada de virtualização.

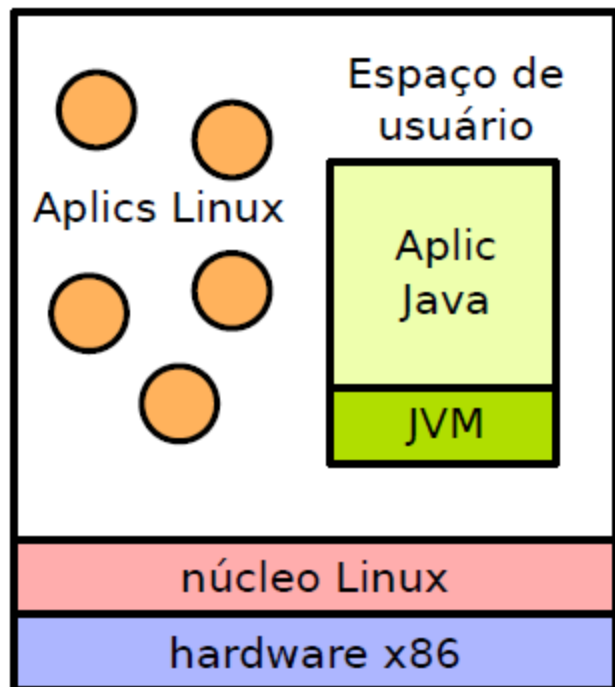
Virtualização - *Histórico*

- Década de 1970:
 - Pesquisadores definem os requisitos básicos para que um hardware suporte virtualização.
 - Primeiras experiências concretas na área.
- Década de 1990:
 - Virtualização volta à tona:
 - Aumento do desempenho de PC's / Java
- Atualmente:
 - Várias linguagens são compiladas para VM's
 - HW com suporte nativo à virtualização.

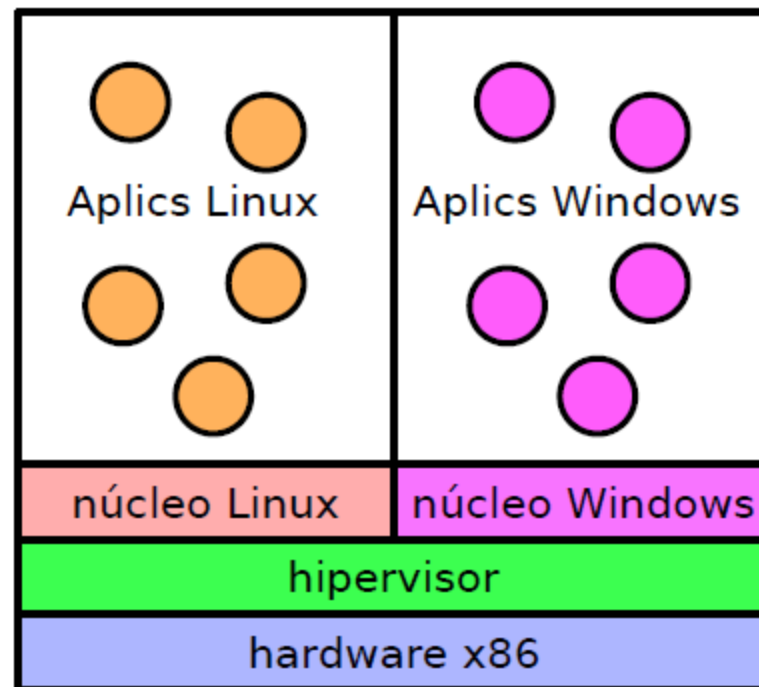
Virtualização - *Implementação*

- Máquina Virtual de Aplicação:
 - Ambientes destinados a suportar apenas um processo ou aplicação convidada específica.
 - Exemplo: *JVM – Java Virtual Machine*
- Máquina Virtual de Sistema:
 - Construídas para suportar Sistemas Operacionais completos
 - Exemplo: *VMWare, Xen, VirtualBox*

Virtualização - *Implementação*



VM de aplicação



VM de sistema

Virtualização – *MV de Aplicação*

- UCSD Pascal (*1970's*)
 - Linguagem pascal compilado para *p-code*.
 - *p-code* era executado pela MV *p-machine*.
- C# (*C sharp*)
 - Linguagem é compilada para CIL (*Common Intermediate Language*).
 - CIL é executado pela MV CLR (*Common Language Runtime*).
- Outras: *Java, Python, Perl, Lua, Ruby*

Virtualização – *MV de Sistema*

- Suportam um ou mais SO's convidados
- Cada SO acha que está sozinho sobre o hardware.
- Softwares podem rodar normalmente sobre os SO's convidados.

Virtualização – *MV de Sistema*

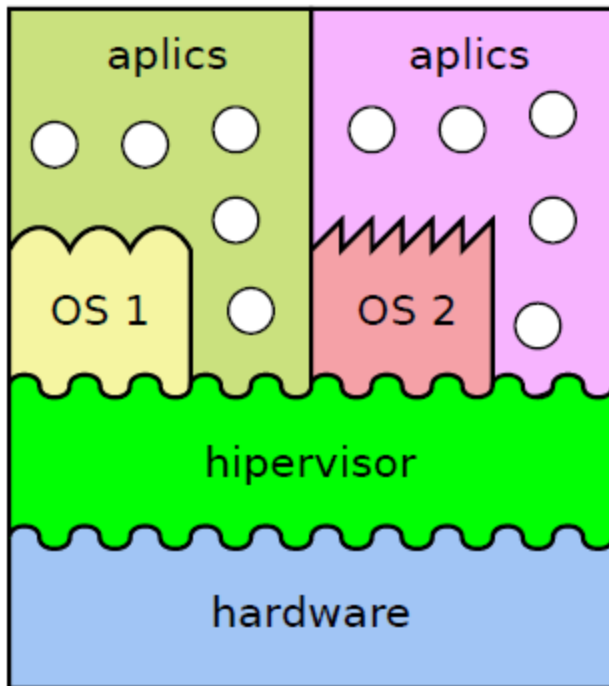
- Monitores Nativos

- Hypervisor executa diretamente sobre o hardware.
- Tem como função gerenciar os recursos de hardware entre os SO's convidados.

- Monitores Convidados

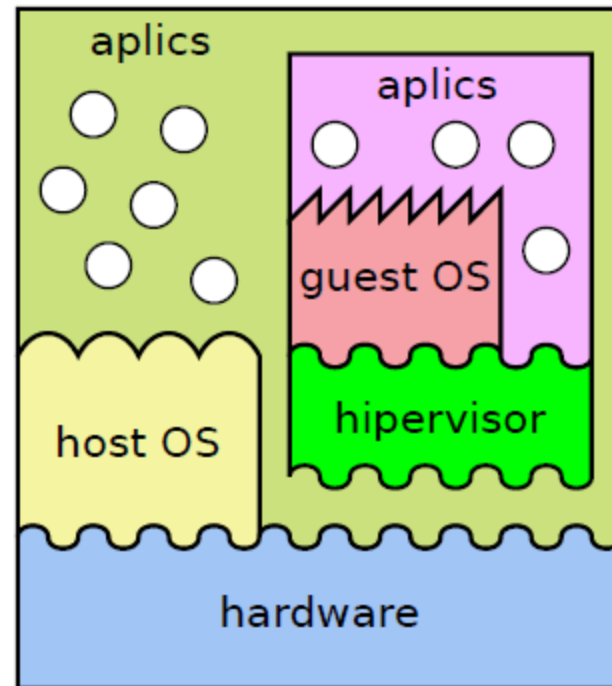
- Hypervisor executa sobre um SO hospedeiro
- Usa recursos do SO real para oferecer serviços ao SO convidado.

Virtualização – *MV de Sistema*



hypervisor nativo

*IBM OS/370;
VMWare ESX Server
Xen*



hypervisor convidado

*QEmu;
VMWare Workstation
VirtualBox*

Virtualização - *Vantagens*

- Aperfeiçoamento e testes de novos sistemas operacionais;
- Executar diferentes sistemas operacionais sobre o mesmo hardware, simultaneamente;
- Simular alterações e falhas no hardware para testes ou reconfiguração de um SO, provendo confiabilidade e escalabilidade para as aplicações;
- Garantir a portabilidade das aplicações legadas (que executariam sobre uma VM simulando o sistema operacional original);
- Diminuir custos com hardware.

Virtualização - *Desvantagens*

- Processos menos eficientes em máquinas mais antigas.
- MV é um software, portanto, está sujeito aos problemas de segurança de um software.
- Gerenciamento ainda é um problema embora as ferramentas para instanciar, monitorar, configurar e salvar tenham melhorado muito.